3rd ASSIGNMENT – Physical Design on Oracle® DB

The DBA is in charge, among other responsibilities, of monitoring and tuning the system. In this assignment, we are going to simulate this situation, although we will significantly simplify the processes. Specifically, we will focus on the physical design of a small subset of the system.

3.1 Analysis of the Current State

We will start off the database that we have designed and used in the other two assignments. Specifically, we'll take the proposed solution for the first assignment (which was also the start point for second assignment).

3.1.1 Starting Scenario

The first step is to set up the testing environment. To do this, everything from the second assignment that may hinder the evaluation (views, indexes, triggers, auxiliary tables... and testing data) will be eliminated. Thus, we should destroy the DB and run again the scripts that create and populate it (NEW_creation and NEW_load.sql) and that were provided for the second assignment. To make load more realistic, the following line must be included before final commit in NEW_load.sql:

```
update copies set condition=substr('NGWVD', dbms_random.value(1,6),1);
```

This is the initial scenario, though the scripts can be modified (and run again) during this assignment.

3.1.2 Workload

The first step that we should take is to analyze what operations occur in the database, focusing on the most relevant ones and measuring their frequency. Such analysis has already been carried out, concluding that the most important operations are the four following queries:

```
select * from editions where pub_place='…';
-- values like 'Madrid', 'Segovia', or 'Barataria', to name a few examples
select * from editions where publisher='…';
select * from copies where condition='…';
select * from editions;
```

The four queries show these frequencies, respectively: {0.3, 0.3, 0.3, 0.1}. With the former SQL code, and taking into account those frequencies, a workload has been created that includes the operations to be run and in the proper proportion and order.

3.1.3 Performance metrics

Along with this statement, a script (script_statistics_2025.sql) is provided. When run, it creates in the database a new package (PKG_COSTES), containing two procedures that implement the workload and make testing possible. The PR_WORKLOAD procedure contains the workload itself; and the procedure RUN_TEST is used for repeating the workload one or more times (indicated by the *ite* parameter) and displaying the average cost (in both accesses and response time).

Therefore, the next step is to study the performance of the standard workload to see the initial costs. To do this, run the mentioned package creation (this is required only once), and call the *run_test(n)* procedure (it includes a loop for repeat testing *n* times; 10 times is fine; while you could ask for more, consider the current load of the server and avoid saturating it with too many iterations).

3.2 Tunning Proposals: new physical design

Once we know the efficiency of the initial design, the next step is to analyze current state and its possibilities. Specifically, we will analyze (a) the initial physical design; (b) the nature of the operations to be executed; (c) the execution plan of each operation (algorithms used in its resolution), providing details of its execution; and (d) the average costs calculated over several runs.

Based on this study, we will develop our physical design proposal aimed at reducing the number of accesses to secondary storage, and the global running time. Physical design proposals may entail changes that require rebuilding part of the system. With already existing systems, structural changes are difficult to implement, because they usually involve altering structures in production that already contain data. Conversely, newly created systems and testing environments ease the application of structural changes.

In this assignment, almost any structure studied in the subject is allowed (if it is within the possibilities of the DBMS, of course). You can change physical parameters of the objects (pctfree, tablespace, etc.), create structures (single-table or multi-table clusters, indexes, etc.), add execution directives (hints), ... However, materialized views and bitmap indexes won't be allowed. It is mandatory to consider (at least) the inclusion of one index and one cluster. Each element's design, implementation and evaluation have to be included in the assignment report; nevertheless, if when analyzing its performance it is concluded that it is not advantageous, you can remove it from the final design (keeping its design and performance measurements in the report).

The physical design should be implemented along with the original structures (that is, modifying the original script *NEW_creation.sql* script). When running it, everything will be destroyed and created again with the new physical design, although empty of course. After executing it, the database must be populated again by running *NEW_load.sql.* In case the workload is modified (due to the inclusion of running hints), it will also be necessary to create the package PKG_COSTES again, by running the (modified) script.

3.3 Final Evaluation (of the new physical design)

The last step would be to verify that the improvements have had the desired effect. The results of the execution of the workload on the initial physical design must be compared with the results obtained on the complete physical design proposed and implemented. Optionally, the analysis can be extended by performing a new physical design and evaluating it: usually, the process of monitoring and tuning the system is a process of continuous refinement, although in this practice this is not necessary (it's only intended to improve the starting off situation in some way, and to provide a good analysis). You can (optionally) modify the experiment to evaluate the performance of successive executions without the restart operation (of the DB state) being performed.

Summary of steps to take:

- **Starting situation**: restore the original state of the DB
- **Measure** the performance of the standard workload (both time and number of accesses) executing several times.
- **Analyze** the execution plan for each operation, extracting the processes (typology) of which it is composed. Classify and sort all the processes in the physical structure, their frequency and an approximation of the cost.
- Based on that analysis, develop a **proposal** to reduce the number of accesses to secondary memory.
- Describe the **complete physical design**, starting from the basic organizations involved in it, as well as the auxiliary structures used to improve the performance.
- **Implement** the proposed complete physical design, and measure its performance over the proposed standard workload.
- **Compare** the performance of the original physical design versus the proposed one and interpret the results. Optionally, you can refine the proposal and implement improvements to the physical design (whose performance you should analyze and compare with the previous ones).
- **Document** all the work done in this assignment, paying special attention to physical design, performance measurements, execution plans, improvement proposals, and comparison of results.

Documentation to deliver:

A report must be submitted containing all the steps described above. In addition, it will contain the code for implementing the complete physical design that you have made (not all the code; only the modifications included into the DB creation script, or into the package).

Supporting Materials:

- Slides corresponding to the lab session.
- SQL Scripts (Oracle PL/SQL syntax): standard workload and procedures related to performance measurements based on querying the statistics collected and stored by Oracle.
- Oracle DBMS 12c user account.